Wordnet

Wordnet is a library and database of the python nltk. It allows lookup of definitions, different types of synonyms, and more. It is useful because you can find out how workds are linked.

In [2]:
```
import nltk
from nltk.corpus import wordnet as wn
print(wn.synsets('dragon'))
```

```
[Synset('dragon.n.01'), Synset('dragon.n.02'), Synset('draco.n.02'), Synset('drago
n.n.04')]
```

In [12]:
```
dragon = wn.synset('dragon.n.01')
print(dragon.definition())
print(dragon.examples())
print(dragon.lemmas())
hyper = dragon.hypernyms()[0]
top = wn.synset("entity.n.01")
while(hyper):
    print(hyper)
    if hyper == top:
        break
    if hyper.hypernyms():
        hyper = hyper.hypernyms()[0]
```

```
a creature of Teutonic mythology; usually represented as breathing fire and having
a reptilian body and sometimes wings
[]
[Lemma('dragon.n.01.dragon'), Lemma('dragon.n.01.firedrake')]
Synset('mythical_monster.n.01')
Synset('monster.n.01')
Synset('imaginary_being.n.01')
Synset('imagination.n.01')
Synset('creativity.n.01')
Synset('ability.n.02')
Synset('cognition.n.01')
Synset('psychological_feature.n.01')
Synset('abstraction.n.06')
Synset('entity.n.01')
```

Wordnet's nouns are all hyponyms of the noun entity. Also noticed that dragon's hypernyms were mainly concepts rather than creatures.

In [24]:
```
print(dragon.hypernyms())
print(dragon.hyponyms())
print(dragon.part_meronyms())
print(dragon.part_holonyms())
print(dragon.lemmas()[0].antonyms())
```

```
[Synset('mythical_monster.n.01')]
[Synset('wyvern.n.01')]
[]
[]
[]
```

In [19]:
```
print(wn.synsets('bother'))
```

```
[Synset('fuss.n.02'), Synset('annoyance.n.04'), Synset('trouble_oneself.v.01'), Syn
set('annoy.v.01'), Synset('trouble.v.02'), Synset('bother.v.04'), Synset('bother.v.
05'), Synset('bother.v.06')]
```

In [21]:
```
bother = wn.synset('bother.v.04')
print(bother.definition())
print(bother.examples())
print(bother.lemmas())
hyper = bother.hypernyms()[0]
top = wn.synset("enter.v.01")
while(hyper):
    print(hyper)
    if hyper == top:
        break
    if hyper.hypernyms():
        hyper = hyper.hypernyms()[0]
```

```
intrude or enter uninvited
["Don't bother the professor while she is grading term papers"]
[Lemma('bother.v.04.bother')]
Synset('intrude.v.01')
Synset('enter.v.01')
```

I noticed that for verbs there are fewer thing and they aren't all under the same thing like nouns. Of course, that could be because I chose a more obscure verb than I did noun.

In [29]:
```
print(wn.morphy("bother", wn.NOUN))
print(wn.morphy("bother", wn.ADJ))
print(wn.morphy("bother", wn.VERB))
```

```
bother
None
bother
```

In [3]:
```
from nltk.wsd import lesk
rake = wn.synset("rake.n.03")
claw = wn.synset("claw.n.03")
print(wn.wup_similarity(rake,claw))
test = ["the", "rake", "looked", "like", "a", "claw", "."]
print(lesk(test, "rake", "n"))
print(lesk(test, "claw", "n"))
```

```
0.23529411764705882
Synset('rake.n.03')
Synset('claw.n.03')
```

Rake and claw are more similar than I expected despite being concepually only similar in shape and non-noun synonyms. They are only slightly lower than hit and slap despite being a lot less similar as non-noun synonyms.

SentiWordNet

SentiWordNet is short for sentimental word net. It gets word sentiments, which is where words are positively, neutrally or negitively connotated. Word senitments can become sentence sentiments.

In [15]:
```
from nltk.corpus import sentiwordnet as swn

barf = swn.senti_synset("sick.a.01")
print(barf)
sentiSent = ["you", "make", "me", "sick"]
for ss in sentiSent:
    sslist = list(swn.senti_synsets(ss))
    if(sslist):
        print(ss)
        print(sslist[0].neg_score())
        print(sslist[0].pos_score())
```

```
<ill.a.01: PosScore=0.125 NegScore=0.75>
make
0.0
0.5
me
0.0
0.0
sick
0.0
0.0
```

Sick as an adjective is mostly negative, althought there is a slight positive. "you make me sick" is a very negative sentence, but analising each word on its own without context gives a net positve sentiment.

Collocations

A collocation is a pair of words that mean more together than individually. Sick and tired imply emotional exausition while sick only impies physical illness and tired implies another sort of physical exaustion.

In [31]:
```
from nltk.book import text4
import math
text4.collocations()
txt = ' '.join(text4.tokens)
print(math.log2(txt.count('Vice President') * len(text4.tokens)/(txt.count('Vice')
```

```
United States; fellow citizens; years ago; four years; Federal
Government; General Government; American people; Vice President; God
bless; Chief Justice; one another; fellow Americans; Old World;
Almighty God; Fellow citizens; Chief Magistrate; every citizen; Indian
tribes; public debt; foreign nations
10.389348302486308
```

Vice President is a collocation because both words add to the meaning, specifying a specific position different from president, and not related to the usual use of vice. It's not suprising it's very strongly a collocation.